**HTTP:**

*HTTP* facilitates exchange of information on the World Wide Web by enabling **communication between clients and servers** over the Internet

The first release of *HTTP 0.9* enabled only clients to request information from a server using a **single operation: GET**. The first HTTP release only supported transmitting **ASCII data**. However, in the following releases, this support expanded to other data types.

*HTTP 1.0* provides **two new methods** (POST and HEAD). This made HTTP responses to contain a **status code**, thus enabling the receiver to check the request processing status (successful or failed). HTTP header enabled clients to send and receive **different file types** and exchange relevant metadata

HTTP 1.1:

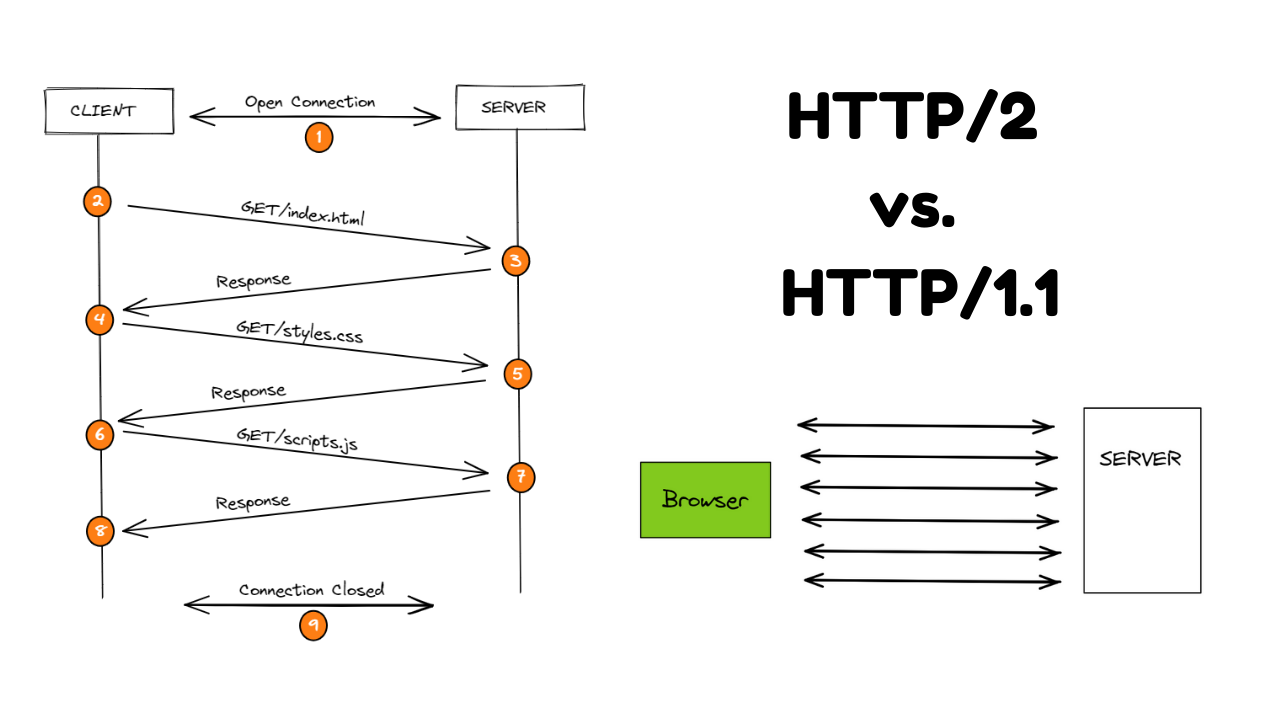
1. 1.1 version added **six extra methods**: PUT, PATCH, DELETE, CONNECT, TRACE, and OPTIONS
2. In addition, compression and decompression, multi-language support, and byte-range transfers are all introduced

However we had few limitations here and are solved in next release HTTP 2.0

HTTP2.0:

HTTP 2.0 focused on improving the **protocol performance**. It implemented several features to improve connections and data exchange. Let’s see some of them:

1. **Request multiplexing**: HTTP 1.1 is a sequential protocol. So, we can send a single request at a time. HTTP 2.0, in turn, allows sending requests and receiving responses asynchronously. In this way, we can do multiple requests at the same time using a single connection
2. **Request prioritization**: with HTTP 2.0, we can set a numeric prioritization in a batch of requests. Thus, we can be explicit in which order we expect the responses, such as getting a webpage CSS before its JS files
3. **Automatic compressing**: in the previous version of HTTP (1.1), we must explicitly require the compression of requests and responses. HTTP 2.0, however, executes a GZip compression automatically
4. **Connection reset**: a functionality that allows closing a connection between a server and a client for some reason, thus immediately opening a new one
5. **Server push**: to avoid a server receiving lots of requests, HTTP 2.0 introduced server push functionality. With that, the server tries to predict the resources that will be requested soon. So, the server proactively pushes these resources to the client cache



**JavaScript Objects:**

Objects, in JavaScript, are the most important data type and form the building blocks for modern JavaScript. These objects are quite different from JavaScript’s primitive data types (Number, String, Boolean, null, undefined, and symbol) in the sense that these primitive data types all store a single value each (depending on their types).

* Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.
* An object is a **reference data type**. Variables that are assigned a reference value are given a reference or a pointer to that value. That reference or pointer points to the location in memory where the object is stored. The variables don’t actually store the value.
* Objects in JavaScript may be defined as an unordered collection of related data, of primitive or reference types, in the form of “**key: value**” pairs. These keys can be variables or functions and are called properties and methods, respectively.